LAMP 系统性能调优，第 3 部分

内容

MySQL 服务器调优

概览

利用服务器的几个调优技巧，让 MySQL 服务器飞速运行

关于 MySQL 调优

记录慢速查询

Sean Walberg

对查询进行缓存 2004 年 7 月 30 日发布

强制限制

缓冲区和缓存

3 个必不可少的工具

结束语

相关主题

系列内容：

此内容是该系列 3 部分中的第 3 部分： **LAMP 系统性能调优**

关于 MySQL 调优

有 3 种方法可以加快 MySQL 服务器的运行速度，效率从低到高依次为：

1. 替换有问题的硬件。
2. 对 MySQL 进程的设置进行调优。
3. 对查询进行优化。

替换有问题的硬件通常是我们的第一考虑，主要原因是数据库会占用大量资源。不过这种解决方

案也就仅限于此了。实际上，您通常可以让中央处理器（CPU）或磁盘速度加倍，也可以让内存增大 4 到 8 倍。

第二种方法是对 MySQL 服务器（也称为 mysqld）进行调优。对这个进程进行调优意味着适当地分配内存，并让 mysqld 了解将会承受何种类型的负载。加快磁盘运行速度不如减少所需的磁盘访问次数。类似地，确保 MySQL 进程正确操作就意味着它花费在服务查询上的时间要多于花费在处理后台任务（如处理临时磁盘表或打开和关闭文件）上的时间。对 mysqld 进行调优是本文的重点。

最好的方法是确保查询已经进行了优化。这意味着对表应用了适当的索引，查询是按照可以充分的。尽管本文并没有包含查询调优方面的内容（很多著作中已经针对这个主题进行了探讨），不需要进行调优的查询。

内容

概览

虽然已经为这些任务指派了次序，但是仍然要注意硬件和 mysqld 的设置以利于适当地调优查询。见过速度很快的机器在运行设计良好的查询时由于负载过重而失败，因为 mysqld 被大量繁忙的

记录慢速查询

在一个 SQL 服务器中，数据表都是保存在磁盘上的。索引为服务器提供了一种在表中查找特定表、*表*的*行*必须要搜索整个表时，就称为*表扫描*。通常来说，您可能只希望获得表中数据的一个子集盘 I/O，因此也就会浪费大量时间。当必须对数据进行连接时，这个问题就更加复杂了，因为必比较。

当然，表扫描并不总是会带来问题；有时读取整个表反而会比从中挑选出一部分数据更加有效出这些决定）。如果索引的使用效率很低，或者根本就不能使用索引，则会减慢查询速度，而且增加，这个问题会变得更加显著。执行时间超过给定时间范围的查询就称为*慢速查询*。

您可以配置 mysqld 将这些慢速查询记录到适当命名的慢速查询日志中。管理员然后会查看这个有哪些部分需要进一步调查。清单 1 给出了要启用慢速查询日志需要在 my.cnf 中所做的配置。

清单 1. 启用 MySQL 慢速查询日志

```
[mysqld]
; enable the slow query log, default 10 seconds
log-slow-queries
; log queries taking longer than 5 seconds
long_query_time = 5
; log queries that don't use indexes even if they take less than long_query_ti
; MySQL 4.1 and newer only
```

```
8 log-queries-not-using-indexes
```

这三个设置一起使用，可以记录执行时间超过 5 秒和没有使用索引的查询。请注意有关 log-qu
告：您必须使用 MySQL 4.1 或更高版本。慢速查询日志都保存在 MySQL 数据目录中，名为 *ho*
一个不同的名字或路径，可以在 my.cnf 中使用 `log-slow-queries = /new/path/to/file` 实

阅读慢速查询日志最好是通过 `mysqldumpslow` 命令进行。指定日志文件的路径，就可以看到一
还显示了它们在日志文件中出现的次数。一个非常有用的特性是 `mysqldumpslow` 在比较结果之
因此对同一个查询的不同调用被计为一次；这可以帮助找出需要工作量最多的查询。

内容

对查询进行缓存

概览

关于对查询进行缓存

很多 LAMP 应用程序都严重依赖于数据库，但却会反复执行相同的查询。每次执行查询时，数
对查询进行分析，确定如何执行查询，从磁盘中加载信息，然后将结果返回给客户机。MySQL
记录慢速查询
（后面会用到的）查询结果保存在内存中。在很多情况下，这会极大地提高性能。不过，问题
对查询进行缓存
的

强制限制

将 `query_cache_size = 32M` 添加到 /etc/my.conf 中可以启用 32MB 的查询缓存。

缓冲区和缓存

3 个设置参数

监视查询缓存

结束语

在启用查询缓存之后，重要的是要理解它是否得到了有效的使用。MySQL 有几个可以查看的变
相关主题
清单 2 给出了缓存的状态。

评论

清单 2. 显示查询缓存的统计信息

```
1  mysql> SHOW STATUS LIKE 'qcache%';
2  +--------------------------+------------+
3  | Variable_name            | Value      |
4  +--------------------------+------------+
5  | Qcache_free_blocks       | 5216       |
6  | Qcache_free_memory       | 14640664   |
7  | Qcache_hits              | 2581646882 |
8  | Qcache_inserts           | 360210964  |
9  | Qcache_lowmem_prunes     | 281680433  |
10 | Qcache_not_cached        | 79740667   |
11 | Qcache_queries_in_cache  | 16927      |
12 | Qcache_total_blocks      | 47042      |
13 +--------------------------+------------+
```

```
14 | 8 rows in set (0.00 sec)
```

这些项的解释如表 1 所示。

表 1. MySQL 查询缓存变量

变量名	说明
Qcache_free_blocks	缓存中相邻内存块的个数。数目大说明可能有碎片。FLUSH Q 进行整理，从而得到一个空闲块。
Qcache_free_memory	缓存中的空闲内存。
Qcache_hits	每次查询在缓存中命中时就增大。
Qcache_inserts	每次插入一个查询时就增大。命中次数除以插入次数就是不中 率。在上面这个例子中，大约有 87% 的查询都在缓存中命中。
Qcache_lowmem_prunes	缓存出现内存不足并且必须要进行清理以便为更多查询提供空 来看；如果这个数字在不断增长，就表示可能碎片非常严重， free_blocks 和 free_memory 可以告诉您属于哪种情况）。
Qcache_not_cached	不适合进行缓存的查询的数量，通常是由于这些查询不是 SEL
Qcache_queries_in_cache	当前缓存的查询（和响应）的数量。
Qcache_total_blocks	缓存中块的数量。

通常，间隔几秒显示这些变量就可以看出区别，这可以帮助确定缓存是否正在有效地使用。运行 数器，如果服务器已经运行了一段时间，这会非常有帮助。

使用非常大的查询缓存，期望可以缓存所有东西，这种想法非常诱人。由于 mysqld 必须要对缓 低时执行剪除，因此服务器可能会在试图管理缓存时而陷入困境。作为一条规则，如果 FLUSH 那就说明缓存太大了。

强制限制

您可以在 mysqld 中强制一些限制来确保系统负载不会导致资源耗尽的情况出现。清单 3 给出了
设置。

清单 3. MySQL 资源设置

```
1 set-variable=max_connections=500
2 set-variable=wait_timeout=10
3 max_connect_errors = 100
```

内容

连接最大个数是在第一行中进行管理的。与 Apache 中的 MaxClients 类似，其想法是确保只建
服务器上目前建立过的最大连接数，请执行 SHOW STATUS LIKE 'max_used_connections'。

第 2 行告诉 mysqld 终止所有空闲时间超过 10 秒的连接。在 LAMP 应用程序中，连接数据库的
求所花费的时间。有时候，如果负载过重，连接会挂起，并且会占用连接表空间。如果有多个
连接，那么将这个值设低一点并不可取！

最后一行是一个安全的方法。如果一个主机在连接到服务器时有问题，并重试很多次后放弃，
FLUSH HOSTS 之后才能运行。默认情况下，10 次失败就足以导致锁定了。将这个值修改为 10
中恢复。如果重试 100 次都无法建立连接，那么使用再高的值也不会有太多帮助，可能它根本

缓冲区和缓存

MySQL 支持超过 100 个的可调节设置；但是幸运的是，掌握少数几个就可以满足大部分需要。
SHOW STATUS 命令查看状态变量，从中可以确定 mysqld 的运作情况是否符合我们的预期。给缓
系统中的现有内存，因此调优通常都需要进行一些妥协。

MySQL 可调节设置可以应用于整个 mysqld 进程，也可以应用于单个客户机会话。

服务器端的设置

每个表都可以表示为磁盘上的一个文件，必须先打开，后读取。为了加快从文件中读取数据的

了缓存，其最大数目由 /etc/mysqld.conf 中的 table_cache 指定。清单 4 给出了显示与打开表

清单 4. 显示打开表的活动

```
mysql> SHOW STATUS LIKE 'open%tables';
+---------------+-------+
| Variable_name | Value |
+---------------+-------+
| Open_tables   | 5000  |
| Opened_tables | 195   |
+---------------+-------+
2 rows in set (0.00 sec)
```

清单 4 说明目前有 5,000 个表是打开的，有 195 个表需要打开，因为现在缓存中已经没有可用
前面已经清除了，因此可能会存在 5,000 个打开表中只有 195 个打开记录的情况）。如果 Ope
STATUS 命令快速增加，就说明缓存命中率不够。如果 Open_tables 比 table_cache 设置小很
间可以增很总不是什么坏事）。例如，使用 table_cache = 5000 可以调整表的缓存。

与表的缓存类似，对于线程来说也有一个缓存。 mysqld 在接收连接时会根据需要生成线程。在
对查询进行缓存便于以后使用可以加快最初的连接。

清单 5 显示如何确定是否缓存了足够的线程。

清单 5. 显示线程使用统计信息

```
mysql> SHOW STATUS LIKE 'threads%';
+-------------------+--------+
| Variable_name     | Value  |
+-------------------+--------+
| Threads_cached    | 27     |
| Threads_connected | 15     |
| Threads_created   | 838610 |
| Threads_running   | 3      |
+-------------------+--------+
4 rows in set (0.00 sec)
```

此处重要的值是 Threads_created，每次 mysqld 需要创建一个新线程时，这个值都会增加。如
STATUS 命令时快速增加，就应该尝试增大线程缓存。例如，可以在 my.cnf 中使用 thread_cac

关键字缓冲区保存了 MyISAM 表的索引块。理想情况下，对于这些块的请求应该来自于内存，
如何确定有多少块是从磁盘中读取的，以及有多少块是从内存中读取的。

清单 6. 确定关键字效率

```
mysql> show status like '%key_read%';
+-------------------+-----------+
| Variable_name     | Value     |
+-------------------+-----------+
| Key_read_requests | 163554268 |
| Key_reads         | 98247     |
+-------------------+-----------+
2 rows in set (0.00 sec)
```

Key_reads 代表命中磁盘的请求个数， Key_read_requests 是总数。命中磁盘的读请求数除以
本例中每 1,000 个请求，大约有 0.6 个没有命中内存。如果每 1,000 个请求中命中磁盘的数目
缓冲区了。例如，key_buffer = 384M 会将缓冲区设置为 384MB。

临时表可以在更高级的查询中使用，其中数据在进一步进行处理（例如 GROUP BY 字句）之前，
情况下，在内存中创建临时表。但是如果临时表变得太大，就需要写入磁盘中。清单 7 给出了

清单 7. 确定临时表的使用

```
mysql> SHOW STATUS LIKE 'created_tmp%';
+-------------------------+-------+
| Variable_name           | Value |
+-------------------------+-------+
| Created_tmp_disk_tables | 30660 |
| Created_tmp_files       | 2     |
| Created_tmp_tables      | 32912 |
+-------------------------+-------+
3 rows in set (0.00 sec)
```

每次使用临时表都会增大 Created_tmp_tables；基于磁盘的表也会增大 Created_tmp_disk_
么严格的规则，因为这依赖于所涉及的查询。长时间观察 Created_tmp_disk_tables 会显示所
定设置的效率。 tmp_table_size 和 max_heap_table_size 都可以控制临时表的最大大小，因
都进行了设置。

每个会话的设置

下面这些设置针对于每个会话。在设置这些数字时要十分谨慎，因为它们在乘以可能存在的连接
存！您可以通过代码修改会话中的这些数字，或者在 my.cnf 中为所有会话修改这些设置。

当 MySQL 必须要进行排序时，就会在从磁盘上读取数据时分配一个排序缓冲区来存放这些数据
么数据就必须保存到磁盘上的临时文件中，并再次进行排序。如果 sort_merge_passes 状态变
情况。清单 8 给出了一些与排序相关的状态计数器信息。

清单 8. 显示排序统计信息

```
mysql> SHOW STATUS LIKE "sort%";
+-------------------+---------+
| Variable_name     | Value   |
+-------------------+---------+
| Sort_merge_passes | 1       |
| Sort_range        | 79192   |
| Sort_rows         | 2066532 |
| Sort_scan         | 44006   |
+-------------------+---------+
4 rows in set (0.00 sec)
```

内容

概览

关于 MySQL 调优

4MB

记录慢速查询

对查询进行缓存

强制限制

缓冲区和缓存

3 个必不可少的工具

结束语

相关主题

评论

如果 sort_merge_passes 很大，就表示需要注意 sort_buffer_size。例如， sort_buffer_s

MySQL 也会分配一些内存来读取表。理想情况下，索引提供了足够多的信息，可以只读入所需
佳或数据本性使然）需要读取表中大量数据。要理解这种行为，需要知道运行了多少个 SELECT
行数据的次数（而不是通过索引直接访问）。实现这种功能的命令如清单 9 所示。

清单 9. 确定表扫描比率

```
mysql> SHOW STATUS LIKE "com_select";
+---------------+--------+
| Variable_name | Value  |
+---------------+--------+
| Com_select    | 318243 |
+---------------+--------+
1 row in set (0.00 sec)

mysql> SHOW STATUS LIKE "handler_read_rnd_next";
+-----------------------+-----------+
| Variable_name         | Value     |
+-----------------------+-----------+
| Handler_read_rnd_next | 165959471 |
+-----------------------+-----------+
1 row in set (0.00 sec)
```

Handler_read_rnd_next / Com_select 得出了表扫描比率 —— 在本例中是 521:1。如果该值超

read_buffer_size，例如 read_buffer_size = 4M。如果这个数字超过了 8M，就应该与开发
优了！

3 个必不可少的工具

尽管在了解具体设置时，SHOW STATUS 命令会非常有用，但是您还需要一些工具来解释 mysql
个工具是必不可少的；在 参考资料 一节中您可以找到相应的链接。

大部分系统管理员都非常熟悉 top 命令，它为任务所消耗的 CPU 和内存提供了一个不断更新的
真；它为所有连接上的客户机以及它们正在运行的查询提供了一个视图。mytop 还提供了一个
的实时数据和历史数据，以及有关正在运行的查询的统计信息。这是一个很有用的工具，可以
的状况，您可以获得有关服务器健康信息的视图，并显示导致问题的任何连接。

mysqlard 是一个连接到 MySQL 服务器上的守护程序，负责每 5 分钟搜集一次数据，并将它们
Database 中。有一个 Web 页面会显示这些数据，例如表缓存的使用情况、关键字效率、连接
况。尽管 mytop 提供了服务器健康信息的快照，但是 mysqlard 则提供了长期的健康信息。作
的一些信息针对如何对服务器进行调优给出一些建议。

搜集 SHOW STATUS 信息的另外一个工具是 mysqlreport。其报告要远比 mysqlard 更加复杂，
分析。这是对服务器进行调优的一个非常好的工具，因为它对状态变量进行适当计算来帮助

结束语

本文介绍了对 MySQL 进行调优的一些基础知识，并对这个针对 LAMP 组件进行调优的 3 部分系
度上需要理解组件的工作原理，确定它们是否正常工作，进行一些调整，并重新评测。每个组
MySQL —— 都有各种各样的需求。分别理解各个组件可以帮助减少可能会导致应用程序速度变

相关主题

您可以参阅本文在 developerWorks 全球站点上的 英文原文。

“从 MySQL 或 PostgreSQL 迁移到 DB2 Express-C”（developerWorks，2006 年 6 月）提
Express-C 上的简单方法。

IBM 还为那些希望迁移到 DB2 Express-C 上的 MySQL 管理员提供了帮助，请参阅：“利用

DB2 与 MySQL 的管理任务和基本任务”（developerWorks，2006 年 2 月）以及本系列文章

SHOW VARIABLES 和 SHOW STATUS 在 MySQL 文档中都已经很好地进行了定义。

如果喜欢 blogs，MySQL Performance Blog、 Xaprb 以及 MySQL DBA 都非常值得阅读。

在 developerWorks 上 Architecture 专区 中，可以找到提高架构设计领域方面技能所需要的

展 LAMP 应用程序的关键。

尽管已经出版了 3 年之久了，*High Performance MySQL* 仍然是非常有价值的一本书。作者

MySQL 的各种文章。

mytop 告诉您目前 MySQL 服务器上都在进行什么操作，并提供一些关键的统计信息。在发

于这个程序。

内容

mysqlard 会给出 MySQL 服务器一个关键性能指示器的图形表示，并给出一些调优建议。

概览

mysqlreport 是一个必须的工具。它为您分析 SHOW STATUS 变量。

关于 MySQL 调优

MySQL 文章如果没有提供到 phpMyAdmin 的链接，就说不上完整。尽管已经给出了对状态

强大之处在于如何简化管理任务。

记录慢速查询

在 developerWorks Linux 专区 中可找到适合于 Linux 开发人员的更多资源，包括 Linux 教

对查询进行缓存

章和教程。

强制限制

利用可直接从 developerWorks 下载的 IBM 试用软件 在 Linux 上构建您的下一个开发项目。

缓冲区和缓存

评论

3 个必不可少的工具

结束语

添加或订阅评论，请先登录或注册。

相关主题

有新评论时提醒我

评论

IBM **Developer**

站点反馈

我要投稿

报告滥用

第三方提示

关注微博

大学合作

选择语言

English

中文

日本語

Русский

Português (Brasil)

Español

한글

Code patterns

技术文档库

软件下载

开发者中心

订阅源

时事通讯

视频

博客

活动

社区

概览